The Theory is Predictive, but is it Complete? An Application to Human Perception of Randomness

Jon Kleinberg* Annie Liang† Sendhil Mullainathan‡

April 15, 2016

Abstract

When we test a theory using data, it is common to focus on *correctness*: do the predictions of the theory match what we see in the data? But we also care about a property we might call *completeness*: how much of the predictable variation in the data is captured by the theory? This question is difficult to answer, because in general we do not know how much "predictable variation" there is in the problem. This paper proposes the use of machine learning algorithms as a means of constructing a benchmark level for the best attainable level of prediction. We illustrate this approach on the problem of predicting human generation of random sequences. Relative to an atheoretical machine learning algorithm benchmark, we find that existing behavioral models explain roughly 10 to 30% of the predictable variation in this problem. This fraction is robust across several datasets, suggesting that (1) there is a significant amount of structure in this problem that our models have yet to capture and (2) machine learning may provide a generally viable approach to testing theory completeness.

1 Introduction

When we test theories, it is common to focus on what one might call *correctness*: do the predictions of the theory match what we see in the data? For example, we can test a theory that says that wages are determined by one's knowledge and capabilities, by looking at whether more education indeed predicts higher wages in labor data. Such a finding would provide evidence in support of the theory, but little guidance towards whether an alternative theory might fit the data even better. Beyond correctness, we also care about this latter feature, which we will refer to as

*Cornell University
†Harvard University
‡Harvard University

completeness: how much of the explainable variation in the data is captured by the theory?

Measurement of the completeness of our theories is important because it provides guidance on the marginal (predictive) gain of improvement in modeling. If our models are fairly complete, then new theories should not be expected to drastically improve predictive power (they may, of course, aid towards other goals—for example, by providing new conceptual insight into the problem). In contrast, if our models are far from complete, then new models have the potential to lead to large improvements in prediction. Completeness therefore guides both our understanding of the achievements already made by existing models, and also of the potential progress that remains ahead.

Despite an interest in completeness, we focus on correctness in social science for a pragmatic reason. We can measure the fit of any given theory to data, but we have no intuition for what constitutes a "good" fit. For example, suppose we are interested in predicting a binary variable and find that a given theory predicts accurately in 55% of observed trials. Is this achievement significant? For certain problems—e.g. prediction of changes in stock returns given a past history of returns—55% accuracy is a stunning success. In other problems—e.g. prediction of college matriculation given socioeconomic and other personal characteristics—it is only mediocre. This significant variation in predictability across problems means that perfect accuracy is not a universally appropriate benchmark for our theories: we need to understand how well a theory's predictive power lines up against some best *achievable* accuracy.

The purpose of this paper is to propose a practically implementable way to generate this benchmark, via methods in machine learning. As we discuss more fully in Section 5, this is an approach that is also being proposed contemporaneously with our work by Peysakhovich & Naecker (N.d.). Recent advances in machine learning have enabled substantial progress in problems of prediction, but are often criticized for using atheroetical and uninterpretable models, for example by searching for the best prediction function over a large set of explanatory variables. The resulting prediction functions perform well empirically but rarely reveal a deep theoretical structure.

Rather than considering machine learning as a replacement for existing theories, a role for which it (currently) seems ill-suited, our goal is to leverage its techniques towards the alternative goal of assessing theory completeness. The approach we propose is simple: compare the performance of existing (interpretable and economically meaningful) models to the performance of atheoretical machine learning algorithms.

We illustrate this approach on a simple problem with a long history of study in psychology and behavioral economics: human generation of random sequences. It is well documented that humans misperceive randomness (Bar-Hillel & Wagenaar, 1991; Kahneman & Tversky 1972), with implications in many economic settings

(Rabin & Vayanos 2010, Chen, Moskowitz & Shue N.d.). Leading models for human misperception of randomness include Rabin (2002) and Rabin & Vayanos (2010). We are interested in assessing the success of these models towards predicting human generation of fair coin flips.

To this end, we use the platform Mechanical Turk to collect 14,050 strings of length eight, produced as if by flipping a fair coin several times in succession. We ask two questions. First: can we predict the eighth flip in a string if we are given the first seven? Second: can we separate human-generated strings from strings generated by a true Bernoulli(0.5) process in a mixed sample?

We adapt Rabin (2002) and Rabin & Vayanos (2010) for these prediction problems, and find that these models achieve (mean-squared) prediction errors of approximately 0.249. These prediction errors are improvements upon a naive baseline of 0.25 (corresponding to guessing at random). The problem of interest regards the interpretation of the improvement of approximately 0.001 on the naive baseline. How significant is this reduction in prediction error, and how much could we hope to improve upon it? To answer these questions, we need a benchmark for *achievable* prediction error.

There are several ways to construct such a benchmark using machine learning prediction techniques. For our first set of results, we draw on an appealing property of our domain: despite its conceptual richness, it has a compact enough representation that we can construct an essentially "perfect" benchmark based on *table lookup*, an algorithm that uses the empirical distribution of the full set of combinatorially distinct strings in a training set to predict new strings. Using this approach, we achieve prediction errors of approximately 0.243. If we take this to be our benchmark, then existing behavioral models produce roughly 12% of the achievable improvement in prediction error for this problem.

In the remainder of the paper, we outline and respond to two potential critiques of this method for measuring completeness. The first is feasibility. Table lookup can be implemented in the given problem because of the size of the domain space, but is not a generally viable strategy. How sensitive is the estimated measure of completeness to the choice of machine learning algorithm? Towards this concern, we report the the prediction error achieved by two alternative (standard) machine learning approaches: LASSO regression and a decision tree algorithm. The best of these algorithms achieves a prediction error of approximately 0.243, suggesting that the benchmark constructed using table lookup may be approximated by other machine learning algorithms that scale to problems with more complex representations.

A second direction of concern regards whether the estimated ratio is special to the problem of prediction of eight-length strings of coin flips. This may be the case if, for example, table lookup succeeds by capturing specific features of generation of eight-length H/T strings that do not generalize to related problems. We thus

examine the stability of estimated completeness when we change the prediction task to predicting data collected in *related but non-identical* contexts. Specifically, we learn a model for human generation of randomness using the original data of eight-length coin flips, and then use this model to predict strings generated in a nearby domain.

We consider two variations in the domain. In Section 3.1, we relabel the possible outcomes: instead of asking subjects to generate binary strings generated as if repeatedly flipping a fair coin labelled "Heads" and "Tails", we impose new frames in which the coin is either labelled "@" on one side and "!" on the other, or "r" on one side and "2" on the other. In Section 3.2, we change the index of the flip to be predicted: instead of asking subjects to generate eight repeated coin flips and predicting the eighth, we ask subjects to generate fifteen coin flips and try to predict flips 9-15. We find that in these modified prediction problems, the existing models produce between 7-30% of the improvement in prediction error obtained using table lookup, providing evidence that the benchmark and ratio discovered previously are indeed stable across local problem domains.

Taken together, these results suggest that: (1) there is a significant amount of structure in problem of prediction of human generation of randomness that existing models have yet to capture and (2) machine learning may provide a generally viable approach to testing theory completeness.

2 Primary setting: human generation of coin flips

2.1 Description of data

We asked 334 subjects on Mechanical Turk to generate 50 binary strings of length eight, each, as if these strings were the realizations of 50 experiments in which a fair coin was flipped 8 times. The task was described to subjects using the text below:

> We are researchers interested in how well humans can produce randomness. A coin flip, as you know, is about as random as it gets. Your job is to mimic a coin. We will ask you to generate 8 flips of a coin. You are to simply give us a sequence of Heads (H) and Tails (T) just like what we would get if we flipped a coin.
>
> Important: We are interested in how people do at this task. So it is important to us that you not actually flip a coin or use some other randomizing device.

Entry of the coin flips was implemented through eight drop-down menus, each of which had the options "H" and "T". Subject effort was incentivized through the following text:

> To encourage effort in this task, we have developed an algorithm (based on previous Mechanical Turkers) that detects human-generated coin flips from computer-generated coin flips. You are approved for payment only if our computer is not able to identify your flips as human-generated with high confidence.

Additionally, to discourage use of an external randomizing device, we required subjects to complete each string in 30 seconds or less. The complete set of directions can be found in Appendix A.

The "algorithm" that we use for detection of lazy subjects identified the 26 strings whose empirical frequency exceeded the 90th percentile across strings (0.0059). These strings were categorized as *over-generated*, and subjects who produced 20 or more over-generated strings were removed from the data. In total, this criterion removed 53 subjects, or 2900 strings. The remaining dataset consists of a total of 281 unique subjects, or 14,050 unique strings. Throughout, we identify Heads with '1' and Tails with '0,' so that each string is an element of $\{1, 0\}^8$.

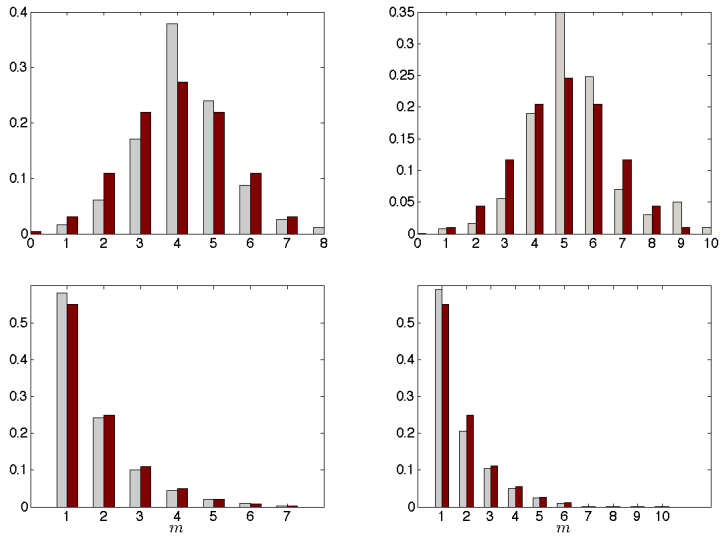


Figure 1: (a) Top row. Distribution of the number of heads in the realized string. *Left:* Comparison of MTurk data with theoretical Bernoulli predictions. *Right:* Comparison of Nickerson & Butler (2009) data with theoretical Bernoulli predictions. (b) Bottom row. Distribution of proportion of runs which are of length m. *Left:* Comparison of MTurk data with theoretical Bernoulli predictions. *Right:* Comparison of Nickerson & Butler (2009) data with theoretical Bernoulli predictions.

We find that the observed distribution over strings is unlikely to have been generated by a true Bernoulli(0.5) process: the hypothesis that the true distribution over $\{1, 0\}^8$ is uniform is rejected under a χ^2 test with $p \approx 0$. Moreover, the nature

of mis-generation is qualitatively consistent with comparative references (we use Nickerson & Butler (2009) and Rapaport & Budescu (1997)). For example, there is an over-tendency towards alternation (52.16% of flips are different from the previous flip, as compared to an expected 50% in a Bernoulli(0.5) process), an under-tendency to generate strings with "extreme" ratios of Heads to Tails (see the top row of Figure 6), and an under-tendency to generate strings with long runs (see the bottom row of Figure 6).

Additionally, subjects display strong context-dependency: the probability of reversal depends on several previous flips. Table 1 compares this dependence in our data with statistics reported in Rabin & Vayanos (2010) (using data from Rapaport & Budescu (1997)), listing the respective probabilities that various three-flip patterns are followed by '1.' Except for a much softer contrast in our data between the probability with which '000′ and '111′ are followed by '1', we find that these conditional probabilities are quite similar.

			Our data	Rapaport & Budescu (1997)	Bernoulli
0	1	0	0.5995	0.588	0.5
1	0	0	0.5406	0.62	0.5
0	0	1	0.5189	0.513	0.5
0	0	0	0.5185	0.70	0.5
1	1	1	0.4811	0.30	0.5
0	1	1	0.4595	0.38	0.5
1	1	0	0.4528	0.487	0.5
1	0	1	0.4415	0.412	0.5

Table 1: The empirical probability of Heads, conditional on three fixed previous flips: (1) the actual proportion of generated Heads in our data, (2) the assessed probability of Heads next flip from Rapaport & Budescu (1997), as presented in Rabin and Vayanos (2010), (3) probabilities consistent with a Bernoulli(0.5) process.

2.2 Theories of misperception

Motivated by empirical findings such as those described above, several frameworks have been proposed for modeling human misperception of randomness. We consider in particular the two approaches proposed in Rabin (2002) and Rabin & Vayanos (2010).

Rabin (2002) models subjects who observe i.i.d. signals, but mistakenly believe them to be negatively autocorrelated. Specifically, subjects observe a sequence of i.i.d. draws from a Bernoulli(θ) distribution, where $\theta \in [0, 1]$ is an unknown rate drawn from distribution π. Although subjects know the correct distribution π, they have a mistaken belief about the way in which the realized rate θ determines

the signal process. Subjects believe that the observed signals are drawn *without replacement* from an urn containing θN '1' signals and $(1-\theta)N$ '0' signals, so that a signal of '1' is less likely following observation of '0', and vice versa. For convenience, the author imposes an additional, stylized, assumption in which subjects believe that the urn is "refreshed" every other round, meaning that the composition is returned to θN '1' signals and $(1-\theta)N$ '0' signals.

This model is primarily intended as a model of mistaken inference, and not directly as a model of generation of random sequences, so a few adaptations are needed to carry this model into our setting. We alter the model in the following ways: First, since subjects are told the bias of the coin (fair), we fix the distribution π over rates so subjects know that $\theta = 0.5$ with certainty. Second, we relax the assumption that the urn is refreshed deterministically every other round, adding in a second parameter $p \in [0,1]$, which determines the probability that the urn is refreshed. In this revised model, subjects generate random sequences by drawing without replacement from an urn that is initially composed of $0.5N$ '1' balls and $0.5N$ '0' balls, and is subsequently refreshed with probability p before every draw.

The model proposed in Rabin & Vayanos (2010) is similar in spirit, but richer. We use the following version of their model, which is closest to our setting. Each subject generates the first flip, s_1, according to a true Bernoulli(0.5) distribution. Then, each subsequent flip, s_k, is determined according to

$$s_k \sim \text{Ber}\left(0.5 - \alpha \sum_{k=0}^{7} \delta^k (2s_{7-k} - 1)\right)$$

where the constant $\delta \in [0,1]$ captures a (decaying) influence of past flips, and the constant $\alpha \geq 0$ measures the strength of negative autocorrelation. Notice that $2s_k - 1 = 1$ if $s_k = 1$ and $2s_k - 1 = -1$ if $s_k = 0$, so that past instances of '1' reduce the probability that the k-th flip is '1', and past instances of '0' increase this probability.

2.3 Prediction tasks

We test these theories by looking at how well they predict features of our data. We focus on two tests in particular. In the first test, which we refer to as *continuation*, we try to guess a subject's eighth flip from his first seven flips. A prediction rule for this problem is any map

$$f : \{0,1\}^7 \to [0,1]$$

that takes 7-length strings into the probability that the eighth flip is '1'. The error in predicting a dataset $\{s^i\}_{i=1}^n$ of n strings is measured using mean-squared error:

$$\mathcal{L}(f) = \sum_{i=1}^{n} \left(s_8^i - f(s_{1:7}^i)\right)^2.$$

To gain some intuition on the possible values of $\mathcal{L}(f)$, let us briefly note the following.

Fact 1. *Suppose that strings in* $\{s^i\}_{i=1}^n$ *are generated i.i.d. from a Bernoulli(0.5) distribution. Then,* $\mathbb{E}[\mathcal{L}(f)] \geq 0.25$ *for every possible prediction rule* f*.*

That is, if subjects are truly generating strings according to a Bernoulli(0.5) process, then we cannot improve on an expected mean-squared error of 0.25. If, instead, subjects are generating strings according to either of the behavioral models described above, then we can do better by leveraging the first seven flips. Optimal prediction rules (in the sense of minimizing expected mean-squared loss) are defined for each of these models in Appendix B, and denoted f_R and f_{RV} respectively.

In the second test, which we refer to as *classification*, we are presented with a dataset of strings—half generated by human subjects, and half generated by a Bernoulli(0.5) process—and seek to separate the human-generated strings from the computer-generated strings. A prediction rule in this problem is any map

$$c : \{0,1\}^8 \to [0,1]$$

from eight-length strings into a probability that the string was generated by a human subject. The error in predicting a dataset $\{s^i\}_{i=1}^n$ of n strings is measured using mean-squared error

$$\mathcal{L}(c) = \sum_{i=1}^{n} \left(c^i - c\left(s^i\right)\right)^2,$$

where $c^i = 1$ if the true source of generation for string s^i was a human subject, and $c^i = 0$ otherwise.[1] In an abuse of notation, we use $\mathcal{L}$ to refer to the loss function in both problems, trusting that no confusion will arise. As above, we have the following.

Fact 2. *Suppose that strings in* $\{s^i\}_{i=1}^n$ *are generated i.i.d. from a Bernoulli(0.5) distribution. Then,* $\mathbb{E}[\mathcal{L}(c)] \geq 0.25$ *for every possible prediction rule* c*.*

Thus, if strings are truly generated according to a Bernoulli(0.5) process, then we cannot improve on an expected prediction error of 0.25. If, instead, the strings are generated according to either behavioral model above, then we can improve upon this error. We define the optimal prediction rules (in the sense of minimizing expected mean-squared loss) for these models in Appendix B, and refer to them (respectively) as c_R and c_{RV}.

[1] A brief comment on the relationship between these two prediction tasks. One may wonder whether success on one implies success on another. This need not be so. Observe that success on the continuation task is achieved by correctly assessing the likelihood of $s_{1:7}1$ versus $s_{1:7}0$. If this ratio is close to the true ratio for every $s_{1:7}$, the unconditional probability of either string $s_{1:7}1$ or $s_{1:7}0$ occurring can be very off.

Following, we test prediction rules f_R, f_{RV} in the continuation task on the Mechanical Turk data, and prediction rules c_R, c_{RV} in the classification test, given a merged dataset consisting of the Mechanical Turk data and an equal number of strings generated according to a Bernoulli(0.5) process. The reported prediction error is obtained using ten-fold cross validation: we (randomly) partition the data into 10 equally-sized subsets, estimate the free parameters of the model on nine subsets (the training set), and predict the strings in the tenth (test set). The reported error is an average across choices of test set.

	Continuation	Classification
Guessing 50-50	0.25	0.25
Rabin (2002)	0.2495	0.2493
	(0.0001)	(0.0001)
Rabin and Vayanos (2010)	0.2491	0.2495
	(0.0001)	(0.0001)

Table 2: Prediction errors achieved using Rabin (2002) and Rabin (2010) are improvements on the prediction error achieved by guessing at random. How do we assess the size of this improvement?

In Table 2, we compare the obtained prediction errors with a naive baseline in which we predict '1' with probability 0.5 in the continuation task, and classify each string as human generated with probability 0.5 in the classification task. The core motivation for this paper is clearly seen here: Both behavioral models are more predictive than the naive baseline, but the improvement on the naive baseline of (up to) 0.0009 is extremely difficult to interpret. How much have these models improved upon the naive prediction rule, and how much could we further hope to improve upon it? To answer these questions, we need a benchmark for obtainable prediction error that is much more suitable than 0.

2.4 Establishing a benchmark

As we discussed in the introduction, our proposed benchmark is the prediction error achieved using a technique we refer to as *table lookup*.

Definition 1 (Table Lookup)**.** *Let g be the empirical distribution over strings in the training data. The table lookup continuation rule is*

$$\hat{f}_T(s_{1:7}) = \frac{g(s_{1:7}1)}{g(s_{1:7})} \quad \textit{for all } s_{1:7} \in \{1,0\}^7. \tag{1}$$

where '$s_{1:7}1$' is the concatenation of strings $s_{1:7}$ and '1.' The table lookup classification rule is $\hat{c}_T(s) = g(s)$.

In the continuation task, the table lookup prediction rule assigns to every string $s_{1:7} \in \{1,0\}^7$ the empirical frequency with which the string is followed by '1' in the training data. In the classification task, the table lookup prediction rule assigns to every string its empirical frequency in the training data. Under the assumption that strings are i.i.d. across subjects, the prediction error achieved using this rule approaches the "lowest possible" prediction error with sufficient training data. This is equivalent to the irreducible error in the problem, or the Bayes error rate.

Table 2.4 compares the prediction error achieved using the behavioral models above with the prediction error achieved using table lookup. As before, prediction error is evaluated using 10-fold cross validation.

We find that table lookup achieves a prediction error of 0.2425 in the continuation task and 0.2430 in the classification task. These errors are far from 0, so this is a nontrivial modification from a benchmark of no error. A simple measure of "completeness" of the existing theories is the ratio of improvement in prediction error achieved by the best behavioral model (over the naive approach) to the improvement in prediction error achieved by table lookup. For example, in the continuation task,

$$\text{ratio of improvement} = \frac{0.25 - \min(0.2495, 0.2491)}{0.25 - 0.2425} = 0.1233$$

and in the classification task,

$$\text{ratio of improvement} = \frac{0.25 - \min(0.2494, 0.2495)}{0.25 - 0.2430} = 0.0857.$$

Taking table lookup as our benchmark for obtainable prediction error, these results suggest that existing behavioral models produce between 9-12% of the achievable improvement in prediction error.

	Continuation	Classification
Bernoulli	0.25	0.25
Rabin (2002)	0.2495	0.2494
	(0.0001)	(0.0001)
Rabin and Vayanos (2010)	0.2491	0.2495
	(0.0001)	(0.0001)
Table Lookup	0.2425	0.2430
	(0.0001)	(0.0001)
Completeness using TL as a benchmark	0.1233	0.0857

Table 3: Comparison of prediction error achieved using behavioral models with prediction error achieved using table lookup. The behavioral models explain between 9-12% of the explainable variation in the data.

2.5 Other possible benchmarks

Table lookup is feasible in this problem because of the size of the domain space (2^7 unique strings in the continuation task, and 2^8 unique strings in the classification task). We can in fact search the space of predictive models to optimality, an approach that will not be feasible in every problem. Can we use other (practically implementable) machine learning algorithms as surrogate benchmarks in these other cases?

In this section, we consider use of two alternative benchmarks. First, we use LASSO regression to select a prediction rule that depends only a small set of covariates. The set of features we consider are:

- the empirical frequency of alternations (the probability that flip s_k is followed by its opposite, averaged across all k)
- indicators for the existence of runs of length $2, 3, \ldots, 8$ in the string
- the total number of '1's in the string
- the index for the first occurrence of '1' in the string

and their interactions up to degree 3. This yields a total of 176 features (including the intercept). Let $x(s) \in \mathbb{R}^{176}$ denote the feature vector corresponding to string s, and define prediction rules

$$f_\beta(s_{1:7}) = x(s_{1:7})^T\beta \quad \text{for all } s_{1:7} \in \{1,0\}^7$$
$$c_\beta(s) = x(s)^T\beta \quad \text{for all } s \in \{1,0\}^8$$

Then, in the continuation problem, the LASSO coefficient vector β solves

$$\underset{\beta\in\mathbb{R}^{176}}{\operatorname{argmin}} \sum_{i=1}^{N} \mathcal{L}(f_\beta) + \lambda\|\beta\|_1,$$

where $\|x\|_1$ denotes the l_1 norm of the vector β (the sum of the absolute values of its components). In the classification problem, the LASSO coefficient vector β solves

$$\underset{\beta\in\mathbb{R}^{176}}{\operatorname{argmin}} \sum_{i=1}^{N} \mathcal{L}(c_\beta) + \lambda\|\beta\|_1.$$

Second, we implement a decision tree algorithm using the same feature set.[2] We consider the class of decision trees in which each node considers a single feature,

[2]Decision trees are a recursive partitioning of the domain space. Formally, a decision tree is a rooted tree, in which each node splits the domain space into two or more subspaces according to a discrete function of the input values.

and each branch corresponds to a different value (or set of values) for the feature. New inputs are classified by proceeding down the decision tree to a terminal node, which determines the class.

Table 2.5 below summarizes the prediction errors achieved using LASSO and decision trees in both problems, and reports also the measure of completeness, using these alternative prediction errors as benchmarks.

	Continuation	Classification
Table lookup	0.2425	0.2430
Completeness using TL as a benchmark	0.1233	0.0857
LASSO	0.2460	0.2484
Completeness using LASSO as a benchmark	0.225	0.375
Decision tree	0.2419	0.2434
Completeness using decision trees as a benchmark	0.1111	0.9091

Table 4: Comparison of prediction error achieved using behavioral models with prediction error achieved using table lookup.

The LASSO prediction rules yield a (tenfold cross-validated) prediction error of 0.2460 in the continuation problem and 0.2484 in the classification problem. Decision trees yield a (tenfold cross-validated) prediction error of 0.2419 in the continuation problem and 0.2434 in the classification problem.[3] Additionally, we find that the estimated measure of completeness varies from 11% to 23% in the continuation task, and from 9% to 37% in the classification task, depending on the choice of algorithm. These results suggest that the benchmark constructed using table lookup may be approximated by other machine learning algorithms that scale to problems with more complex representations.

3 Transfer across Domains

The previous sections established a benchmark error of roughly 0.24, and discovered that existing behavioral models achieve approximately 10% of the possible improvement in prediction error (above a naive baseline). How special are these results to the particular setting that we considered? Should we interpret this benchmark and measure of completeness as pertaining only to human generation of eight-length H/T strings, or do they express a more general truth about the predictability of human generation of random sequences, and the extent to which our existing models have attained this?

[3]The slight improvement of decision trees upon table lookup should be interpreted as noise due to the finite size of our training data.

One reason to be concerned is the possibility that behavioral models capture fundamental aspects of human generation of random sequences (not special to production of fair coin flips), while table lookup relies on specific features of generation of eight-length H/T strings that do not generalize to related problems. For example, 57% of strings begin with '1'. The prediction rules derived from Rabin (2002) and Rabin & Vayanos (2010) don't leverage this feature for prediction, but table lookup does. If the predictive accuracy achieved by table lookup is due in large part to use of features like this, then we should not expect its performance to generalize.

To address this question, we consider two robustness checks of "transfer prediction" across different framings of the generation problem. Our approach is as follows. We first estimate the free parameters of all the models (table lookup, and the two behavioral models) on the original dataset of eight-length H/T strings. Then, we use the estimated models to predict new strings, which are not only *out-of-sample* (not used in the estimation of the free parameters), but also produced in a *modified problem domain* (the generation problem is framed differently). By looking at how well the estimated models predict in this new environment, we can assess whether the features used in table lookup are stable features of misperception across these various domains, or whether they are specific to the original problem.

We consider two variations in the domain. In Section 3.1, we relabel the possible outcomes: instead of asking subjects to generate binary strings generated as if repeatedly flipping a fair coin labelled "Heads" and "Tails", we impose new frames in which the coin is either labelled "@" on one side and "!" on the other, or "r" on one side and "2" on the other. In Section 3.2, we change the index of the flip to be predicted: instead of asking subjects to generate eight repeated coin flips and predicting the eighth, we ask subjects to generate fifteen coin flips and try to predict flips 9-15. We find that in these modified prediction problems, the existing models produce between 7-30% of the improvement in prediction error obtained using table lookup, providing evidence that the benchmark and ratio discovered previously are indeed stable across local problem domains.

3.1 Prediction of New Alphabets

In the first transfer prediction task, we attempt to predict strings under a relabelling of the outcome space from {Heads, Tails} to $\{r, 2\}$, and to $\{@, !\}$. Specifically, we ask 124 subjects on Mechanical Turk to generate 50 binary strings of length eight "as if these strings were the realizations of 50 experiments in which a fair coin labeled 'r' on one side and '2' on another was flipped 8 times." We ask another 114 subjects to generate 50 binary strings of length eight "as if these strings were the realizations of 50 experiments in which a fair coin labeled '@' on one side and '!' on another was flipped 8 times".

Following the procedure outlined in Section 2.2, we determine which strings have an empirical frequency exceeding the 90th percentile, and remove all subjects who produced 20 or more such strings. We also identify strings with elements of $\{1, 0\}$, mapping 'r' and '@' into '1', and '2' and '!' into '0.' We refer to these datasets of strings respectively as D_{r2} and $D_{@!}$, and the original data as D_{HT}.

The prediction problems we examine are the following. First: Suppose we know the first seven flips that a subject generated in D_{r2} ($D_{@!}$). How well can we predict his eighth flip, using only the strings in D_{HT} to train our prediction model? This is the transfer analogue of the continuation task described in Section 2.4.

Second: Suppose we have a dataset combining the strings in D_{r2} ($D_{@!}$) with an equal number of strings generated by a Bernoulli(0.5) process. How well can we separate the human-generated strings from the computer-generated strings, using only the strings in D_{HT} to train our prediction model? This is the transfer analogue of the classification task described in Section 2.4.

To answer these questions, we estimate the free parameters in Rabin (2002), Rabin & Vayanos (2010), and table lookup using D_{HT}, and then use the estimated models to predict strings in D_{r2} and $D_{@!}$. The resulting prediction errors (reported below as ten-fold cross validated errors) are listed in Table 3.2, as well as a measure of completeness, using the table lookup prediction error as a benchmark.

	Continuation		Classification	
	$\{r, 2\}$	$\{@, !\}$	$\{r, 2\}$	$\{@, !\}$
Guessing 50-50	0.25	0.25	0.25	0.25
Rabin (2002)	0.2493	0.2499	0.2499	0.2493
	(0.0001)	(0.0001)	(0.0001)	(0.0001)
Rabin and Vayanos (2010)	0.2491	0.2497	0.2491	0.2501
	(0.0001)	(0.0001)	(0.0001)	(0.0001)
Table Lookup	0.2451	0.2456	0.2431	0.2434
	(0.0001)	(0.0001)	(0.0001)	(0.0001)
Completeness using TL as a benchmark	0.1836	0.0682	0.1304	0.1061

Table 5: We train table lookup and our two behavioral models on the original 8-length $\{H, T\}$ data, and then use the estimated data to predict 8-length $\{r, 2\}$ and $\{@, !\}$ data. Reported prediction errors are tenfold cross-validated mean squared errors.

The most important components of the table above are the following. First, we find that the benchmarks of 0.2425 and 0.2430 discovered previously are incredibly robust across the different framings: table lookup achieves prediction errors ranging from 0.2431 to 0.2456. The easure of completeness ranges between 7% and 18%, and is comparable to the range of nine to 12% found previously.

3.2 Prediction of Subsequent Flips

In the second transfer prediction task, we use the original eight-length strings to predict strings of length fifteen. The data to be predicted was produced by asking 120 subjects on Mechanical Turk to generate 25 binary strings of length fifteen "as if these strings were the realizations of 25 experiments in which a fair coin was flipped 15 times". From this data, we construct seven "ghost" datasets of eight-length strings, each including only flips k through $k+7$, where $k \in \{2, \ldots, 8\}$.

Following the procedure outlined in Section 2.2, we identify the strings whose empirical frequency exceeded the 90th percentile, and remove all subjects who produced 20 or more such strings. We again identify strings with elements of $\{1, 0\}$, mapping 'H' into '1', and 'T' into '0.' The datasets are labeled $D_{k:k+7}$, with $k = 2, \ldots, 8$.

The prediction problems we examine are the following. First: Suppose we know the first seven flips that a subject generated in $D_{k:k+7}$. How well can we predict the final flip, using only the strings in D_{HT} to train our prediction model? This is the transfer analogue of the continuation task described in Section 2.4.

Second: Suppose we have a dataset combining the strings in $D_{k:k+7}$ with an equal number of strings generated by a Bernoulli(0.5) process. How well can we separate these strings into those that are human-generated and computer-generated, using only the strings in D_{HT} to train our prediction model? This is the transfer analogue of the classification task described in Section 2.4.

To answer these questions, we estimate the free parameters in Rabin (2002), Rabin & Vayanos (2010), and table lookup using D_{HT}, and then use the estimated models to predict strings in $D_{k:k+7}$, $k = 2, \ldots, 8$. Table 3.2 reports prediction errors obtained using table lookup, and the two behavioral models. We show two results: first, the prediction error obtained in dataset $D_{8:15}$ alone; second, the prediction error averaged across the seven datasets $D_{k:k+7}$, $k = 2, \ldots, 8$.

We find that the table lookup prediction error ranges from 0.2369 to 0.2421, which is comparable to the range from 0.2425 to 0.2430 discovered earlier, and that the measure of completeness ranges from 27% to 31%, greater but comparable to the previous range of nine to 12%.

4 Discussion

4.1 Guarantees on the benchmark

The problem analyzed in this paper can be reformulated as follows. Let $X = \{X_k\}_{k \geq 1}$ be the "human-generated" $\{1, 0\}$-valued random process. If the conditional distribution of X_8 given the realizations of $X_1, \ldots, X_7$ is non-degenerate, then the 8th flip cannot be predicted perfectly from the first seven (and likewise, realizations of X cannot be perfectly separated from realizations of a true Bernoulli(0.5)

	Continuation		Classification	
	Last flip	Average	Last flip	Average
Guessing 50-50	0.25	0.25	0.25	0.25
Rabin (2002)	0.2500	0.2484	0.2474	0.2479
	(0.0001)	(0.0001)	(0.0001)	(0.0001)
Rabin and Vayanos (2010)	0.2462	0.2466	0.2484	0.2485
	(0.0001)	(0.0001)	(0.0001)	(0.0001)
Table Lookup	0.2369	0.2391	0.2409	0.2421
	(0.0001)	(0.0001)	(0.0001)	(0.0001)
Completeness using TL as a benchmark	0.2900	0.3098	0.2857	0.2772

Table 6: We train table lookup and our two behavioral models on the original 8-length $\{H, T\}$ data, and then use the estimated data to predict the data in $\{D_{k:k+7}\}_{k=2}^{8}$. Reported prediction errors are tenfold cross-validated mean squared errors.

sequence). We want to compare the prediction error achieved using existing models not with 0, but with the *irreducible error*

$$\mathbb{E}\left(X_8 - f^*(X_1, \ldots, X_7)\right)^2 \tag{2}$$

where the expectation is with respect to the distribution over $\{1, 0\}^8$ induced by $\{X_1, \ldots, X_8\}$, and f^* is defined by

$$f^*(x_1, \ldots, x_7) = \Pr(X_8 = 1 | X_1 = x_1, \ldots, X_7 = x_7)$$

This is the error that would be achieved by using the true model X to predict realizations of X_8, known in the literature as the *Bayes risk*. Notice that no function $f : \{0, 1\}^7 \to [0, 1]$ can improve upon f^* in an expected mean-squared error sense.

The prediction error obtained using table lookup is a consistent estimator of the Bayes risk: as the quantity of training data approach infinity, the prediction error achieved using f_{TL} approximates (2) to arbitrary precision. However, this approach, which relies on nonparametric estimation of each $\Pr(1 \,|\, x_1, \ldots, x_7)$ for every $(x_1, \ldots, x_7) \in \{1, 0\}^7$, is not feasible in problems where the domain space is substantially larger. In these more general settings, approaches such as those considered in Section 2.5 (LASSO regression and decision trees) are more viable alternatives. Choice of which algorithm is used to generate the benchmark should rely on domain knowledge regarding the assumptions the process is likely to satisfy. We refer the reader to a large literature on estimation of Bayes risk for theoretical guarantees of different approaches.

4.2 Covariates

Throughout this paper, we have considered a fixed set of explanatory covariates, namely the initial seven flips in the first prediction task, and the string itself in the second. The notion of completeness we have proposed is more precisely stated as a measure of completeness *for this given set of covariates*.

The proposed measure of completeness, therefore, is not instructive towards which additional covariates (beyond initial flips) might be added to improve prediction, or how much prediction accuracy can be improved by adding these covariates. What it does allow us to do is separate two potential reasons for imperfect prediction: low predictive accuracy because we haven't yet identified the most predictive covariates, and low predictive accuracy because our model is using good covariates in a poor way. For example, suppose we find that benchmark accuracy is low (say, 55%). This indicates that we may gain from investigating additional possible covariates, rather than try to improve the current classifier without changing the feature set. If instead, benchmark accuracy is high (say, 80%), while our achieved accuracy is low (say, 55%), then there are large gains potentially to be had in prediction, without the addition of any new features.

4.3 Transfer learning

The core of Section 3 is a question of a transfer learning: how much can we improve prediction of strings generated in a given domain, using knowledge of how strings are generated in a related domain? We focused on transfer learning across human-generated Bernoulli(0.5) sequences with different outcome spaces and string lengths, and found that prediction of strings generated for a given outcome space and string length could be substantially improved using data on strings generated for a different outcome space and a different string length. This is reassuring: it suggests that transfer learning is possible across close problems.

But a fuller understanding of the generalizability of machine learning approaches will require investigation into the extent to which transfer learning is possible. For example, let us consider a larger class of random processes, in which the probability of '1' varies, or the size of the outcome space varies. Can we transfer learn across these more substantive changes in the domain? Concretely: might we use human-generated Bernoulli(0.5) strings to predict human-generated Bernoulli(0.6) strings, or human-generated Bernoulli(0.5) strings to predict human-generated Normal(0,1) strings? We leave this large body of questions for future work.

5 Relationship to Literature

The closest paper to ours is Peysakhovich & Naecker (N.d.), which independently proposes the use of machine learning algorithms to provide a benchmark on obtainable predictive accuracy. The authors focus in this paper on the domain of choice under uncertainty, assessing the ability of classical models to predict choices over (risky and ambiguous) lotteries. They use regularized regression as a benchmark, and find that classical models explain a larger fraction of achievable prediction error in the domain of risk than in the domain of ambiguity.

The question considered in this paper is also distantly related to classical work concerning the learnability of a random process. For example, Jackson, Kalai & Smorodinsky (1999) derive a Bayesian representation for a stochastic process, with the property that component distributions are fine enough to be "sufficient for prediction," but coarse enough so that the components are "learnable." These questions consider *asymptotic* properties of the random process, whereas we focus on learning finite properties of the random process (e.g. the eighth flip of the coin).

Finally, this paper is related to the extensive experimental (Bar-Hillel & Wagenaar 1991, Rapaport & Budescu 1997, Rath 1966, Edwards N.d., Nickerson & Butler 2009, Wagenaar 1972), empirical (Camerer 1989, Chen, Moskowitz & Shue N.d., Gillovich, Vallone & Tversky 1985, Croson & Sundali 2005), and theoretical (Falk & Konald 1997, Tversky & Kahneman 1971, Barberis, Shleifer & Vishny 1998, Rabin & Vayanos 2010) literature on human misperception of randomness.

6 Conclusion

Machine learning has produced techniques that have enabled substantial progress in various academic disciplines. The question of how these techniques are best leveraged for advancing research in the social sciences remains open. The purpose of this paper is to propose one potential use of these techniques: towards assessing theory "completeness." We illustrate this proposal on the simple problem of predicting human generation of fair coin flips. We show that the prediction error obtained using the algorithm "table lookup" is a useful benchmark for evaluation of the success of existing behavioral models—and in particular, that existing models explain up to 30% of the predictable variation in the problem. Moreover, this benchmark is robust across related problems, suggesting that machine learning may provide a generally viable approach to testing theory completeness.

Appendix A: Experiment Instructions

Subjects were presented with the following introduction screen:

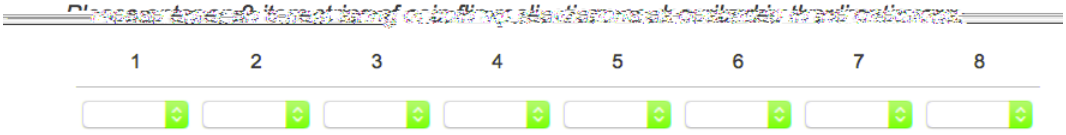
How random can you be?

The challenge.
We are researchers interested in how well humans can produce randomness. A coin flip, as you know, is about as random as it gets. Your job is to mimic a coin. We will ask you to generate 8 flips of a coin. You are to simply give us a sequence of Heads (H) and Tails (T) just like what we would get if we flipped a coin.

Important: We are interested in how people do at this task. So it is important to us that you not actually flip a coin or use some other randomizing device.

How you provide your answer.

You will see a dropdown menu with 8 entries, like this:



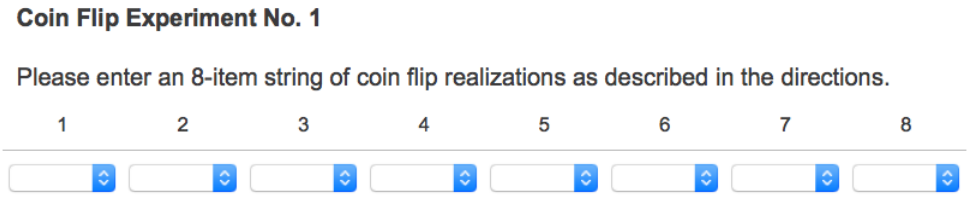
Simply enter the outcome of the first flip under "1", the outcome of the 2nd flip under "2", and so on.

A few tips: instead of choosing an alternative from the dropdown menu, you may input H or T directly from your keyboard. Additionally, you may use the "Tab" key to bring you from one entry to the next.

How many rounds, and how long per round?
There are a total of 50 rounds, and you will have 30 seconds to complete each round. Once your time is up, the question will automatically advance. All questions must be complete for approval for payment.

How is my pay determined?
To encourage effort in this task, we have developed an algorithm (based on previous Mechanical Turkers) that detects human-generated coin flips from computer-generated coin flips. **You are approved for payment only if our computer is not able to identify your flips as human-generated with high confidence.**

Following a trial round and provision of consent, subjects were presented with 50 identical screens that looked like the following:



Subjects were given 30 seconds to complete each string, and a timer displayed their remaining time.

Appendix B: Behavioral Prediction Rules

Rabin (2002) prediction rules. Define continuation rule

$$\hat{f}_R(s_{1:7}) = p(0.5) + \sum_{k=0}^{6} p(1-p)^k \frac{0.5N - \sum_{j=7-k}^{7} s_k}{N}$$

and classification rule

$$\hat{c}_R(s) = \sum_{r \in \{0,1\}^8} \left(p^{\sum r_i} (1-p)^{8-\sum r_i} \right) q(s|r)$$

where

$$q(s|r) = 0.5 r_k + (1 - r_k) \left(\frac{0.5N - \sum_{j=1}^{\min\ j\,:\, r_{k-j}=1} r_{k-j} \mathbb{1}(s_{k-j} = s_k)}{N} \right)$$

is the probability that string s is generated when the urn is refreshed at every '1' in r. There are two free parameters: $p \in [0,1]$ and $N \in \mathbb{N}$.

Rabin & Vayanos (2010) prediction rules. Define prediction rule

$$\hat{f}_{RV}(s_{1:7}) = 0.5 - \alpha \sum_{k=0}^{7} \delta^k (2 s_{7-k} - 1).$$

Define classification rule

$$\hat{c}_{RV}(s) = \sum_k s_k \left(0.5 - \alpha \sum_{j \le k} \delta^{k-j} g(s_j) \right) + (1 - s_k) \left(0.5 + \alpha \sum_{j \le k} \delta^{k-j} g(s_j) \right).$$

There are two free parameters: $\delta \in [0,1]$ and $\alpha \in \mathbb{R}_+$.

References

Bar-Hillel, Maya & Willem Wagenaar. 1991. "The Perception of Randomness." *Advances in Applied Mathematics* .

Barberis, Nicholas, Andrei Shleifer & Robert Vishny. 1998. "A Model of Investor Sentiment." *Journal of Financial Economics* .

Camerer, Colin. 1989. "Does the Basketball Market Believe in the 'Hot Hand'?" *American Economic Review* .

Chen, Daniel, Tobias Moskowitz & Kelly Shue. N.d. "Decision-Making Under the Gambler's Fallacy: Evidence from Asylum Judges, Loan Officers, and Baseball Umpires." Working Paper.

Croson, R. & J. Sundali. 2005. "The Gambler's Fallacy and the Hot Hand: Empirical Data from Casinos." *Journal of Risk and Uncertainty* .

Edwards, W. N.d. "Probability Learning in 1000 Trials." *Journal of Experimental Psychology*. Forthcoming.

Falk, Ruma & Clifford Konald. 1997. "Making Sense of Randomness: Implicit Encoding as a Basis for Judgment." *Psychological Review* .

Gillovich, T., R. Vallone & A. Tversky. 1985. "The Hot Hand in Basketball: On the Misperception of Random Sequences." *Cognitive Psychology* .

Jackson, Matthew O., Ehud Kalai & Rann Smorodinsky. 1999. "Bayesian Representation of Stochastic Processes Under Learning: De Finetti Revisited." *Econometrica* .

Nickerson, Raymond & Susan Butler. 2009. "On Producing Random Sequences." *American Journal of Psychology* .

Peysakhovich, Alexander & Jeffrey Naecker. N.d. "Evaluating Models of Choice under Risk and Ambiguity using Methods from Machine Learning." Working Paper.

Rabin, Matthew. 2002. "Inference by Believers in the Law of Small Numbers." *The Quarterly Journal of Economics* .

Rabin, Matthew & Dmitri Vayanos. 2010. "The Gambler's and Hot-Hand Fallacies: Theory and Applications." *Review of Economic Studies* .

Rapaport, A. & D. Budescu. 1997. "Randomization in Individual Choice Behavior." *Psychological Review* .

Rath, Gustave. 1966. "Randomization by Humans." *The American Journal of Psychology* .

Tversky, Amos & Daniel Kahneman. 1971. "The Belief in the Law of Small Numbers." *Psychological Bulletin* .

Wagenaar, Willem. 1972. "Generation of Random Sequences by Human Subjects: A Critical Survey of the Literature." *Psychological Bulletin* .